**Configuraciones**

Al activar mostrar categorías mostrara categorías y subcategorías

**Categorias**

**Productos**

Para ingresar un producto escoger: categoría, subcategoria.

Si en las configuraciones se activó sin categoría entonces se ingresara las subcategorías como si fueran categorías

Tiene numero de serie indica si el producto a ingresar está asociado a una serie

Flag = permite saber si el producto ya esta terminado de crear **1 se terminó** 0 falta crear

Tipo = indica si es producto terminado, Mercaderia y Servicio

Para generar el seed de productos

PS C:\xampp\htdocs\fe> php artisan iseed productfamilies

Created a seed file from table productfamilies

PS C:\xampp\htdocs\fe> php artisan iseed productatributes

Created a seed file from table productatributes

PS C:\xampp\htdocs\fe> php artisan iseed atribute\_productatribute

Created a seed file from table atribute\_productatribute

PS C:\xampp\htdocs\fe> php artisan iseed local\_productatribute

Created a seed file from table local\_productatribute

Códigos para crear

php artisan make:model Imagegeneral -m

php artisan make:livewire Admin/ShoppingList

explicación de código LpaList.php

  $subcategory = Subcategory::with(['productfamilies' => function ($query) {

            $query->whereHas('productatributes', function ($query) {

                $query->whereHas('locales', function ($query) {

                    $query->where('locals.id', 1) // Reemplaza 'locales.id' con el nombre de la columna 'id' en la tabla 'locales'

                          ->where('local\_productatribute.stock', '>', 0); // Asegúrate de usar el nombre correcto de la tabla 'local\_productatribute' y la columna 'stock'

                });

            });

        }])

        ->findOrFail($this->subcategoryId);

        return $subcategory->productfamilies;

1. **Subcategory::with(['productfamilies' => function ($query) { ... }])**: Aquí se define la relación entre **Subcategory** y **Productfamily** utilizando el método **with()** para cargar las **productfamilies** relacionadas. La función anónima dentro de **with()** permite agregar condiciones adicionales a la consulta de las **productfamilies**.
2. **$query->whereHas('productatributes', function ($query) { ... })**: Dentro de la función anónima, se agrega una condición para asegurarse de que las **productfamilies** tengan **productatributes** relacionados. Esto garantiza que solo se obtendrán las **productfamilies** que tienen **productatributes**.
3. **$query->whereHas('locales', function ($query) { ... })**: Dentro de la función anónima, se agrega una condición adicional para asegurarse de que los **productatributes** estén relacionados con **locales**. Esto garantiza que solo se obtendrán las **productfamilies** que tienen **productatributes** relacionados con **locales**.
4. **$query->where('locals.id', 1)**: Aquí se especifica una condición para filtrar los **locales** por su columna **id**. Asegúrate de reemplazar **'locals.id'** con el nombre real de la columna **id** en la tabla **locales** de tu base de datos.
5. **$query->where('local\_productatribute.stock', '>', 0)**: Esta condición filtra los resultados por el campo **stock** en la tabla **local\_productatribute**. Asegúrate de utilizar el nombre correcto de la tabla **local\_productatribute** y la columna **stock** en tu base de datos.
6. **findOrFail($this->subcategoryId)**: Finalmente, se utiliza el método **findOrFail()** para obtener la subcategoría con el **id** especificado en **$this->subcategoryId**. Si la subcategoría no se encuentra, se generará una excepción.
7. **return $subcategory->productfamilies**: Devuelve las **productfamilies** filtradas por las condiciones establecidas en la consulta.

En resumen, este código realiza una consulta compleja utilizando relaciones entre modelos y condiciones adicionales para obtener las **productfamilies** que cumplen con los criterios establecidos en las condiciones de filtro.